REMARKS

Claims 1, 3-5 and 11-14 are pending and under consideration in this application. Claims 1, 11, and 14 are amended herein. Support for the amendments to the claims may be found at page 11, lines 19-23, and in Fig. 7. Claims 6, 8, 9, and 10 are canceled herein without prejudice or disclaimer. Reconsideration is requested based on the foregoing amendment and the following remarks.

Response to Arguments:

The Applicants appreciate the consideration given to their arguments. The Applicants, however, were disappointed to find that their arguments were not found to be persuasive. The Office Action asserts at page 10, lines 12-17, that:

> The Examiner respectfully disagrees because Ims specifies the input properties (i.e. categories or book titles) are used in retrieving available book inventory information (Ims, col. 10, line 51-53). In other words, Ims' input properties are essentially values used in the retrieval of information. Because the input properties of Ims are used to retrieve information (i.e. col. 10 line 55, Ims states getting results with the input properties), they are a condition used in the retrieval of results.

This is submitted to be incorrect. Neither a category name nor a book title, both of which are given as examples of input properties at column 10, lines 51 and 52 of Ims, are equivalent to a retrieval *condition*, whether or not they are used in retrieving available book inventory information. The input properties of Ims, rather, are used to retrieve information, as noted in the Office Action.

Ims himself, in fact, uses the word "condition" differently than the interpretation given to it in the Office Action. Ims, for example, enables writes to be scheduled for processing when particular events occur or when particular system-specific *conditions* are met. In particular, as described at column 19, lines 39-44:

> For applications which are adaptable to this technique, this aspect of the present invention enables better resource utilization to occur by avoiding unnecessary reads to a back-end data source and by enabling writes to be scheduled for processing when particular events occur or when particular system-specific conditions are met.

Thus, Ims defines a condition as something to be met, rather than values used in the retrieval of information. Ims, consequently, which is cited in the Office Action as evidence of skill

in the art, does not support the interpretation of a "retrieval condition" to which the Office Action adheres.

The Office Action goes on to assert at page 11, lines 2 and 3, that:

> Because these properties are stored together in the instance of a bean, they are stored in correlated form.

This is submitted to be without basis. Whether properties are stored *together* or not has no bearing on whether they are stored in correlated or, for that matter, uncorrelated form.

Still, in the interest of compact prosecution only, and not for any reason of patentability, the final clauses of claims 1, 11, and 14 have been amended to recite substantially "if the number of data records updated is not in the fixed range, the update condition setting unit sets the cache update condition such that the number of date records updated fall in the fixed range."

Further consideration is requested.

Objections to the Specification:

The Specification was subjected to for failing to provide antecedent basis for claimed subject matter. Claims 6, 8, 9, and 10, which recited a computer readable medium, have been canceled. Withdrawal of the objection is earnestly solicited.

Objections to the Claims:

Claims 1, 11, and 14 were objected to for various informalities. Claims 1, 11, and 14 were amended in substantial accord with the Examiner's suggestions. The Examiner's suggestions are appreciated. Withdrawal of the objection is earnestly solicited.

Claim Rejections - 35 U.S.C. § 102:

Claims 1, 3-5 and 11-14 were rejected under 35 U.S.C. § 102(b) as anticipated by U.S. Patent No. 6,505,200 to Ims et al. (hereinafter "Ims"). The rejection is traversed to the extent it might apply to the claims as amended. Reconsideration is earnestly solicited.

In the claimed invention, as described at page 11, lines 19-23 of the specification:

> However, if the number of data records updated within the predetermined period is not in the fixed range (No at Step S302), the update interval setting section 34 sets the update interval such that the number of data records updated fall in a specific range, and then the process is terminated.

The fifth clauses of claims 1 and 11, in particular, recite:

> If the number of data records updated is not in the fixed range, the update condition setting unit sets the cache update condition such that the number of date records updated fall in the fixed range.

Ims neither teaches, discloses, nor suggests "if the number of data records updated is not in the fixed range, the update condition setting unit sets the cache update condition such that the number of date records updated fall in the fixed range," as recited in claims 1 and 11. Ims, in fact, mentions no "data records" or "fixed range" at all.

The second clauses of claims 1 and 11 recite:

> A cache memory that stores in a correlated form the retrieval condition and the retrieval result.

Ims neither teaches, discloses, nor suggests "a cache memory that stores in a correlated form the retrieval condition and the retrieval result," as recited in claims 1 and 11. Ims, rather, is automatically *synchronizing* one data store with another data store, even though the two stores may not share a common format. In particular, as described column 4, lines 52-56:

> An object of the present invention is to provide a technique whereby one data store can be automatically synchronized with another data store, even though the two stores may not share a common format.

Since Ims is automatically synchronizing one data store with another data store, even though the two stores may not share a common format, Ims is not setting "a cache memory that stores in a correlated form the retrieval condition and the retrieval result," as recited in claims 1 and 11.

In Ims, moreover, the cached objects *component* provides for storing, and for automatically refreshing, cached objects. In particular, as described at column 9, lines 53-59:

> The present invention uses a "cached objects" component (referred to equivalently herein as a "cache manager") which caches objects that are used by middleware applications to interact with back-end data sources, where the middleware application functions as a surrogate for a requesting client application. The cached objects component provides for storing, and for automatically refreshing, these objects.

Since, in Ims, the cached objects component provides for storing, and for automatically refreshing, cached objects, Ims is not setting "a cache memory that stores in a correlated form the retrieval condition and the retrieval result," as recited in claims 1 and 11.

In Ims, moreover, a cached object stored by the cached objects component has a set of *input* properties and a set of *output* properties, not a "retrieval condition" or a "retrieval result," as recited in claims 1 and 11. In particular, as described at column 10, lines 21-25:

> Each object stored by the cached objects component has a set of input properties and a set of output properties (which might not be set when the object is cached) representing the information to and from the object's corresponding back-end data source.

Since, in Ims, a cached object stored by the cached objects component has a set of input properties and a set of output properties, Ims is not setting "a cache memory that stores in a correlated form the retrieval condition and the retrieval result," as recited in claims 1 and 11.

In Ims, moreover, a cached object includes processing logic which describes how the object interacts with the *back-end* data source, and an execution method which invokes this processing logic, not a "retrieval condition" or a "retrieval result," as recited in claims 1 and 11. In particular, as described at column 10, lines 21-28:

> Further, each cached object preferably includes processing logic which describes how the object interacts with the back-end data source, and an execution method which invokes this processing logic.

Since, in Ims, a cached object may include processing logic which describes how the object interacts with the back-end data source, and an execution method which invokes this processing logic, Ims is not setting "a cache memory that stores in a correlated form the retrieval condition and the retrieval result," as recited in claims 1 and 11.

Finally, in Ims, the input properties and values thereof are used as an index to *locate* objects stored in the cached objects component, not as a "retrieval condition" or a "retrieval result," as recited in claims 1 and 11. It is the stored objects *themselves*, rather, that may be categorized as either read-access (RA) or write-access (WA). In particular, as described at column 10, lines 28-32:

> The input properties and values thereof are used as an index to locate objects stored in the cached objects component, where those stored objects may be categorized as either read-access (RA) or write-access (WA).

Since, in Ims, the input properties and values thereof are used as an index to locate objects stored in the cached objects component, Ims is not setting "a cache memory that stores in a correlated form the retrieval condition and the retrieval result," as recited in claims 1 and 11.

Furthermore, in Ims, a caching policy specifies when the object is to be *periodically* refreshed, such as at a particular time of day; upon occurrence of a specified event; upon an elapsed time since the last refresh, rather than according to a "retrieval condition and the retrieval result," as recited in claims 1 and 11. In particular, as described at column 13, lines 56-62:

> The caching policy of an RA object preferably specifies when the object is to be periodically refreshed. For example, an object may be refreshed at a particular time of day; upon occurrence of a specified event; upon an elapsed time since the last refresh; etc. This refresh process thereby ensures that a relatively up-to-date version of the back-end data is being used when responding to client read requests.

Since, in Ims, a caching policy preferably specifies when the object is to be periodically refreshed, Ims is not setting "a cache memory that stores in a correlated form the retrieval condition and the retrieval result," as recited in claims 1 and 11.

In Ims, moreover, there is retrieval *logic* in the execution script of the cached object, not a "retrieval condition" or a "retrieval result," as recited in claims 1 and 11. In particular, as described at column 14, lines 51-62:

> For example, cached object HAO2 shown at 312 is refreshed by executing the retrieval logic in the execution script of the cached object. This retrieval logic causes a request to be sent 303 to the back-end data source at host 320, which returns 304 a fresh copy of data values to be used for refreshing the cached object (i.e. re-populating the object's output properties) in the cache 300. Whether a cached copy is already cached or a fresh copy must be retrieved is transparent to the requesting application, which operates as though an actual access (shown in FIG. 3A as imaginary access 325) had been made directly to the back-end data source and a copy had been retrieved in real time.

Since, in Ims, a caching policy preferably specifies when the object is to be periodically refreshed, Ims is not setting "a cache memory that stores in a correlated form the retrieval condition and the retrieval result," as recited in claims 1 and 11.

Finally, in Ims, refresh *logic* is stored with or associated with selected replicated read-access objects, and update *logic* is stored with or associated with selected replication write-access objects, not a "retrieval condition" or a "retrieval result," as recited in claims 1 and 11. In particular, as described at column 5, lines 64-67, continuing at column 6, lines 1 and 2:

> Performing the replication may further comprise executing the refresh logic stored with or associated with selected replicated read-access objects for which the

> queued refresh requests are queued, and executing the update logic stored with or associated with selected replication write-access objects for which the queued update requests are queued.

Since, in Ims, refresh logic is stored with or associated with selected replicated read-access objects, and update logic is stored with or associated with selected replication write-access objects, Ims is not setting "a cache memory that stores in a correlated form the retrieval condition and the retrieval result," as recited in claims 1 and 11.

The fourth clauses of claims 1 and 11 recite:

> An update processing unit that reads the retrieval condition from the cache memory upon fulfillment of the cache update condition.

Ims neither teaches, discloses, nor suggests "an update processing unit that reads the retrieval condition from the cache memory upon fulfillment of the cache update condition," as recited in claims 1 and 11. In Ims, rather, there is retrieval *logic* in the execution script of the cached object, not a "retrieval condition," as discussed above.

Moreover, in Ims, the object's update operation is being applied to the *back-end data source* at host 340, not the cached object. In particular, as described at column 16, lines 58-66:

> When the update policy of an object having elements queued in its update queue 352 is triggered (for example, reaching a certain time of day when low-priority batched mode requests are to be processed), the updates from the queue 352 are processed by executing the update script of the corresponding object. This execution causes the object's update operation to be applied 353 to the back-end data source at host 340, by execution of the object's script using the input property values from the queued element.

Since, in Ims, the object's update operation is being applied to the back-end data source at host 340, not the cached object, Ims is not reading "the retrieval condition from the cache memory upon fulfillment of the cache update condition," as recited in claims 1 and 11.

The fifth clauses of claims 1 and 11 recite further:

> The update condition setting unit sets the cache update condition by acquiring data updated within a predetermined period from the database, and determining whether the number of data records is in a fixed range.

Ims neither teaches, discloses, nor suggests setting "the cache update condition by acquiring data updated within a predetermined period from the database, and determining whether the number of data records is in a fixed range," as recited in claims 1 and 11. Determining the

number of updates requested within a particular *time* period does not amount to determining whether the number of data records is in a fixed range, contrary to the assertion at page 5, lines 10, 11, and 12 of the Office Action. In Ims, rather, delayed updates are selected between the hours of 8 a.m. and 5 p.m. because the system is heavily used in that timeframe. In particular, as described column 15, lines 52-59:

> In a particular system, the update mode may be set such that delayed updates are selected between the hours of 8 a.m. and 5 p.m. because the system is heavily used in that timeframe, as an example, while updates which are requested between midnight and 8 a.m. use the synchronous immediate mode and those requested between 5 p.m. and midnight use the asynchronous immediate mode.

Since, in Ims, delayed updates are selected between the hours of 8 a.m. and 5 p.m. because the system is heavily used in that timeframe, Ims is not setting "the cache update condition by acquiring data updated within a predetermined period from the database, and determining whether the number of data records is in a fixed range," as recited in claims 1 and 11.

Ims, moreover, is counting the number of updates requested within a particular time period, and altering the update mode accordingly in order to reduce network traffic. In particular, as described column 15, lines 59-62:

> Or, more complex evaluations may be performed such as counting the number of updates requested within a particular time period, and altering the update mode accordingly in order to reduce network traffic.

Since Ims is counting the number of updates requested within a particular time period, and altering the update mode accordingly in order to reduce network traffic, Ims is not setting "the cache update condition by acquiring data updated within a predetermined period from the database, and determining whether the number of data records is in a fixed range," as recited in claims 1 and 11.

In Ims, moreover, a cached object is refreshed by executing the retrieval logic in the execution script of the cached object. In particular, as described column 14, lines 51-53:

> For example, cached object HAO2 shown at 312 is refreshed by executing the retrieval logic in the execution script of the cached object.

Since, in Ims, cached object is refreshed by executing the retrieval logic in the execution script of the cached object, Ims is not setting "the cache update condition by acquiring data updated within a predetermined period from the database, and determining whether the number of data

records is in a fixed range," as recited in claims 1 and 11.

In Ims, moreover, the retrieval logic causes a request to be sent to the *back-end* data source, which returns a fresh copy of data values to be used for refreshing the cached object (i.e. re-populating the object's output properties) in the cache. In particular, as described column 14, lines 53-57:

> This retrieval logic causes a request to be sent 303 to the back-end data source at host 320, which returns 304 a fresh copy of data values to be used for refreshing the cached object (i.e. re-populating the object's output properties) in the cache 300.

Since, in Ims, the retrieval logic causes a request to be sent to the back-end data source, which returns a fresh copy of data values to be used for refreshing the cached object (i.e. re-populating the object's output properties) in the cache, Ims is not setting "the cache update condition by acquiring data updated within a predetermined period from the database, and determining whether the number of data records is in a fixed range," as recited in claims 1 and 11.

In Ims, moreover, whether a cached copy is already cached or a fresh copy must be retrieved is *transparent* to the requesting application, which operates as though an actual access had been made directly to the back-end data source and a copy had been retrieved in real time. In particular, as described column 14, lines 57-62:

> Whether a cached copy is already cached or a fresh copy must be retrieved is transparent to the requesting application, which operates as though an actual access (shown in FIG. 3A as imaginary access 325) had been made directly to the back-end data source and a copy had been retrieved in real time.

Since, in Ims, whether a cached copy is already cached or a fresh copy must be retrieved is transparent to the requesting application, which operates as though an actual access had been made directly to the back-end data source and a copy had been retrieved in real time, Ims is not setting "the cache update condition by acquiring data updated within a predetermined period from the database, and determining whether the number of data records is in a fixed range," as recited in claims 1 and 11.

Finally, in Ims, the refresh policy of each cached object may be periodically *evaluated* to determine whether the associated cached objects have become invalid. In particular, as described column 14, lines 62-67:

> In an optional feature of this aspect, the refresh policy of each cached object may

> be periodically evaluated (e.g. at periodic time intervals, or in response to predetermined events) to determine whether the associated cached objects have become invalid.

Since, in Ims, the refresh policy of each cached object may be periodically evaluated to determine whether the associated cached objects have become invalid, Ims is not setting "the cache update condition by acquiring data updated within a predetermined period from the database, and determining whether the number of data records is in a fixed range," as recited in claims 1 and 11. Claims 1 and 11 are submitted to be allowable. Withdrawal of the rejection of claims 1 and 11 is earnestly solicited.

Claims 3, 4, and 5 depend from claim 1 and add further distinguishing elements, while claims 12 and 13 depend from claim 11 and add further distinguishing elements. Claims 3, 4, 5, 12, and 13 are thus also submitted to be allowable. Withdrawal of the rejection of claims 3, 4, 5, 12, and 13 is also earnestly solicited.

Claim 14:

> The eighth clause of claim 14 recites:
>
> Setting the cache update condition by acquiring data updated within a predetermined period from the database, and determining whether a number of data records updated within the predetermined period is in a fixed range, and if the number of data records updated is not in the fixed range, the setting includes setting the cache update condition such that the number of date records updated fall in the fixed range.

Ims neither teaches, discloses, nor suggests " setting the cache update condition by acquiring data updated within a predetermined period from the database, and determining whether a number of data records updated within the predetermined period is in a fixed range, and if the number of data records updated is not in the fixed range, the setting includes setting the cache update condition such that the number of date records updated fall in the fixed range," as discussed above with respect to the rejection of claim 1.

> The second clause of claim 14 recites:
>
> Storing a retrieval request received from a terminal that includes a retrieval condition and a retrieval result retrieved using the retrieval request in a correlated form in a cache memory.

Ims neither teaches, discloses, nor suggests " storing a retrieval request received from a terminal that includes a retrieval condition and a retrieval result retrieved using the retrieval

request in a correlated form in a cache memory," as discussed above with respect to the rejection of claim 1.

The fifth clause of claim 14 recites:

> Reading the retrieval condition from the cache memory upon fulfillment of the cache update condition.

Ims neither teaches, discloses, nor suggests "reading the retrieval condition from the cache memory upon fulfillment of the cache update condition," as discussed above with respect to the rejection of claim 1. Claim 14 is thus also submitted to be allowable for at least those reasons discussed above respect to the rejections of claims 1 and 11. Withdrawal of the rejection of claim 14 is earnestly solicited.

Conclusion:

Accordingly, in view of the reasons given above, it is submitted that all of claims 1, 3-5 and 11-14 are allowable over the cited references. Allowance of all claims 1, 3-5 and 11-14 and of this entire application is therefore respectfully requested.

Finally, if there are any formal matters remaining after this response, the Examiner is invited to telephone the undersigned to attend to these matters.

If there are any additional fees associated with filing of this Amendment, please charge the same to our Deposit Account No. 19-3935.

Respectfully submitted,

STAAS & HALSEY LLP

Date: August 27, 2008

By: /Thomas E. McKiernan/
Thomas E. McKiernan
Registration No. 37,889

1201 New York Avenue, N.W., 7th Floor
Washington, D.C. 20005
Telephone: (202) 434-1500
Facsimile: (202) 434-1501